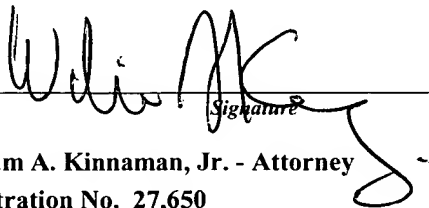| TRANSMITTAL OF APPEAL BRIEF (Large Entity)	Docket No. POU920000026US1

In Re Application Of: George E. Corbin et al

Application No. 09/652,065	Filing Date 8/31/2000	Examiner A. L. Basehoar	Customer No. 33558	Group Art Unit 2178	Confirmation No. 2966

Invention: **Method and Apparatus For Providing Local Data Persistence For Web Applications**

<u>COMMISSIONER FOR PATENTS:</u>

Transmitted herewith in triplicate is the Appeal Brief in this application, with respect to the Notice of Appeal filed on **December 3, 2004**

The fee for filing this Appeal Brief is: **$500.00**

☐   A check in the amount of the fee is enclosed.

☒   The Director has already been authorized to charge fees in this application to a Deposit Account.

☒   The Director is hereby authorized to charge any fees which may be required, or credit any overpayment to Deposit Account No.   **09-0463**

☐   Payment by credit card. Form PTO-2038 is attached.

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

Dated: **January 31, 2005**

William A. Kinnaman, Jr. - Attorney
Registration No. 27,650
IBM Corporation - MS P386
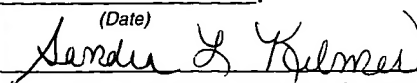**2455 South Road**
Poughkeepsie, NY 12601

Telephone No. (845) 433-1175

RECEIVED

FEB 0 3 2005

U.S. PATENT AND TRADEMARK OFFICE
BOARD OF PATENT APPEALS
AND INTERFERENCES

CC:

P30LARGE/REV06

<div align="right">**Patent**</div>

IN THE U.S. PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Applicant: GEORGE E. CORBIN et al.	:	Group Art Unit: 2178
Serial No.: 09/652,065	:	Examiner: Adam Basehoar
Filed: August 31, 2000	:	January 31, 2005
Confirmation No.: 2966	:	William A. Kinnaman, Jr.
Title: METHOD AND APPARATUS FOR	:	International Business Machines Corporation
PROVIDING LOCAL DATA	:	2455 South Road, Mail Station P386
PERSISTENCE FOR WEB APPLICATIONS	:	Poughkeepsie, NY 12601

<div align="center">**APPLICANTS' APPEAL BRIEF**</div>

Commissioner for Patents

P.O. Box 1450

Alexandria, VA 22313-1450

```
RECEIVED

FEB 0 3 2005

U.S. PATENT AND TRADEMARK OFFICE
BOARD OF PATENT APPEALS
AND INTERFERENCES
```
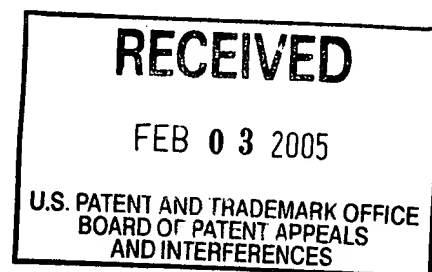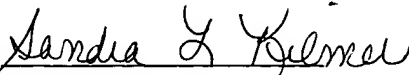
Dear Sir:

Applicants hereby submit their appeal brief in the above-identified application.

<div align="center">**CERTIFICATE OF MAILING UNDER 37 CFR 1.8(a)**</div>

I hereby certify that this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on January 31, 2005.

Sandra L. Kilmer
Sandra L. Kilmer

_1-31-2005_
Date:

REAL PARTY IN INTEREST

The real party in interest is International Business Machines Corporation, the assignee of record.

RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

STATUS OF CLAIMS

Claims 1-17, constituting all pending claims in the application, stand rejected and are on appeal. No claims have been allowed, nor have any claims been cancelled or withdrawn.

STATUS OF AMENDMENTS

A reply after final rejection presenting arguments and formal amendments of the specification, but no amendments of the claims, was filed October 14, 2004. In an advisory action mailed November 17, 2004, applicants were informed that their reply failed to place the application in allowance.

SUMMARY OF CLAIMED SUBJECT MATTER

Claims 1, 6 and 9

Claim 1 is directed to a method of providing local data persistence for a client application (Fig. 1: 108) in an information handling system (102) in which the client application (108) displays (Fig. 3: 300) a first hypertext document (Fig. 9: 902) to a user for entry of user data (as in data entry area 302). The method is performed by the client application 108, which is assumed to have a function (File/Save As) for locally saving displayed documents. In accordance with the invention, the client application (108) receives user data from the user; as well as a save command from the user to save the user data. In response to receiving the save command, the

client application (108) dynamically creates a new hypertext document (Fig. 7: 700) containing the user data and displaying a message (Fig. 4: 402) prompting the user to save the new document (700) using the function (File/Save As) for locally saving displayed documents. The new hypertext document (700) contains a script function (706) that becomes active when the new hypertext document (700) is loaded to perform a desired restoration function.

Claims 6 and 9 are similar to claim 1, but are directed to apparatus and to a program product (specifically, a "program storage device"), respectively.

Claims 13, 15 and 17

Claim 1, dependent on claim 1, further recites that the script function 706 contained in the new hypertext document 700 becomes active when loaded to repopulate the first hypertext document 902 with the previously saved user data (page 11, lines 13-17).

Claims 15 and 17 are similar to claim 13, but depend on apparatus claim 6 and program product claim 9, respectively.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

I.   All claims stand rejected under 35 U.S.C. § 102(b) as being anticipated by the online reference[1] entitled "Persistence" ("WebReference") (paper no. 4, ¶ 2, page 2).

ARGUMENT

Claims 1-12, 14 and 16

This group of claims on appeal contains three independent claims: claims 1, 6 and 9. Claim 1 is representative of this group of claims and reads as follows:

---

[1] This reference may be found at http://www.webreference.com/js/column24/.

1.   In an information handling system in which a client application displays a first hypertext document to a user for entry of user data, said client application having a function for locally saving displayed documents, a method of providing local data persistence for said client application, said method being performed by said client application and comprising the steps of:

receiving user data from said user;

receiving a save command from said user to save said user data; and

in response to receiving said save command, dynamically creating a new hypertext document containing said user data and displaying a message prompting the user to save the new document using said function for locally saving displayed documents, said new hypertext document containing a script function that becomes active when said new hypertext document is loaded to perform a desired restoration function.

Claims 6 and 9 are similar to claim 1, but are directed to apparatus and to a program storage device, respectively.

As noted above, claims 1, 6 and 9 stand rejected under 35 U.S.C. § 102(b) as being anticipated by the online reference[2] entitled "Persistence" ("WebReference") (Final Action, pages 2-3). Since the remaining claims all depend on these base claims, it will be sufficient for the purposes of this appeal to address the Examiner's rejection of these base claims on WebReference. This base rejection is clearly untenable, for the reasons stated below.

In applicants' claimed invention, a <u>first</u> hypertext document—in the specification, frameset document 902 with referenced documents 904 and 906 (Fig. 9)—is displayed to a user for entry of user data, while a <u>new</u> hypertext document—in the specification, save document 700 (Fig. 7)—containing the user data is generated in response to a user command to save the data. The first document can thus be tailored for data entry (Fig. 3), while the second document can be tailored for saving the entered data (Fig. 8).

---

[2] This reference may be found at <u>http://www.webreference.com/js/column24/</u>.

WebReference describes a number of "behaviors" of Microsoft Internet Explorer that allow for data persistence. Perhaps the most relevant such behavior is saveSnapshot, described on the page[3] of WebReference entitled "Hard Disk Persistence". As described in the first paragraph of that page, saveSnapshot is a behavior "that enables persistence of an HTML file when you save it onto your hard disk". The user may save an HTML document containing a partially completed form, using the browser's File/Save As function, and later retrieve the form with the entered data using the File/Open function.

While the saveSnapshot behavior shares some similarities with applicants' claimed invention, it differs fundamentally in that only a single document is involved. Rather than dynamically generating a new document containing user data and displaying a save prompt, the saveSnapshot behavior uses the original document for this purpose. Thus, in the example given for this behavior, the same document (snapshottext.html)[4] is used both to accumulate user data and to store it in persistent form at a selected location on the user's hard drive. The document has the HTML encoding

```
<HTML>
<HEAD>
<META NAME="save" CONTENT="snapshot">
<STYLE>
   .saveSnapshot{behavior:url(#_IE_);}
</STYLE>
</HEAD>
<BODY>
<FORM>Enter a text you want to be persisted: <INPUT TYPE="text"
CLASS="saveSnapshot" ID="inputPersistText"><BR>Enter a text you don't
want to be persisted: <INPUT TYPE="text"><BR><BR>Now save the page as a
Web Page, HTML only. Give a name to the saved Web page. At last, load
the saved Web page. You will see that, as planned, one field is
persistent, while the other one is not.
</FORM>
```
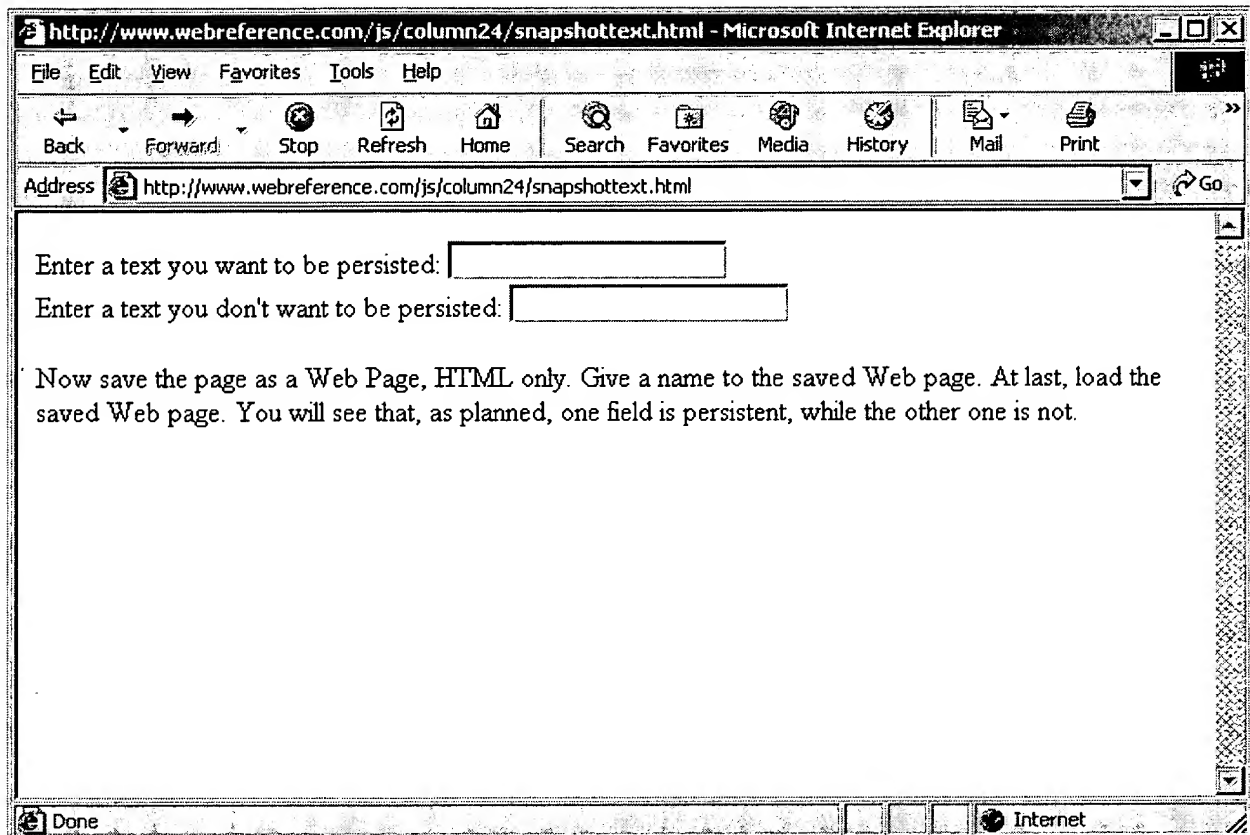
---

[3] This page may be found at http://www.webreference.com/js/column24/snapshot.html.

```
        </BODY>
        </HTML>
```

and appears on the screen as shown below:

Note that this document contains not only a data entry prompt ("Enter a text you want to be persisted:"), but also a save prompt ("Now save the page as a Web Page, HTML only.") and a retrieve prompt ("At last, load the saved Web page") as well. This results in a rather cluttered appearance and contrasts with applicants' scheme, in which the data entry window 300 (Fig. 3) corresponding to frameset document 902 (Fig. 9) is tailored for data entry (with only buttons for Save and Load), while the save window 400 (Fig. 4) corresponding to save document 700 (Fig. 7) is tailored for saving the entered data.

---

[4] This document may be found at http://www.webreference.com/js/column24/snapshottext.html.

The Examiner argues that a "newly created document" is generated when the WebReference user data is "saved locally in persistent form at a selected location on the user's hard drive" (Advisory Action, page 2). However, the document that is saved locally is not a "newly created" document, but the very same document (perhaps with a different file name) that the user is viewing and editing in his browser window. Also, if the act of saving a document is equated with the dynamic creation of a new document claimed by applicants, it seems a little strange having such a "newly created" document display a message prompting the user to save it (as also claimed by applicants), since that has already been done.

The Examiner also argues that the code listings in the saveSnapshot behavior, which are rendered on the screen as shown above, "are not a limiting factor and are merely examples of possible functionality" (Advisory Action, page 2). However, applicants were noting these code listings not as a "limiting factor", but as a consequence of not dynamically creating a new hypertext document as claimed. Because the saveSnapshot behavior does not dynamically create a new hypertext document with the save prompt as claimed by applicants, a person using the saveSnapshot behavior and desiring such a prompt must include it in the original document.

Accordingly, WebReference does not teach dynamically creating a new hypertext document containing user data and displaying a message prompting the user to save the new document using a local save function as claimed by applicants. Therefore, the claims in this group distinguish patentably over WebReference.

Claims 13, 15 and 17

Claims 13, 15 and 17 are dependent on base claims 1, 6 and 9, respectively, and therefore distinguish patentably over WebReference for the reasons noted above with respect to those claims.

Claims 13, 15 and 17 further distinguish patentably over WebReference by virtue of their recitation that the script function contained in the new hypertext document becomes active when loaded to repopulate the first hypertext document with said user data (page 11, lines 13-17).

The Examiner has referred to the script block

```
<SCRIPT CLASS="saveSnapshot" ID="persistentScript">
  var persistentVariable;
</SCRIPT>
```

described on the "Hard Disk Persistence" page. Even assuming, for the sake of argument, that this script block is a script function, it does not repopulate a first document with user data as claimed by applicants. Rather, it only ensures the persistence of certain variables in the same document in which it appears. This script block is thus to be contrasted with the script function saveFields() shown in Appendix D of the specification, which is replete with the names and values of variables to be repopulated to the header file 904.
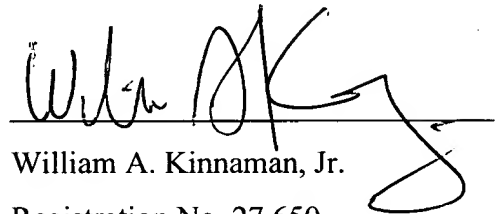
Accordingly, WebReference does not teach providing a script function in a <u>new</u> hypertext document that becomes active when loaded to repopulate a <u>first</u> hypertext document with user data as claimed by applicants. Therefore, the claims in this group as well distinguish patentably over WebReference.

Conclusion

For the foregoing reasons, the Examiner's rejection of claims 1-17 as being anticipated by WebReference is clearly untenable and should be reversed by the Board.

<div style="margin-left:50%">

Respectfully submitted,

GEORGE E. CORBIN et al.

By   William A. Kinnaman, Jr.

William A. Kinnaman, Jr.

Registration No. 27,650

Phone: (845) 433-1175

Fax: (845) 432-9601

</div>

WAK/wak

CLAIMS APPENDIX

Claims on Appeal

1.   In an information handling system in which a client application displays a first hypertext document to a user for entry of user data, said client application having a function for locally saving displayed documents, a method of providing local data persistence for said client application, said method being performed by said client application and comprising the steps of:

receiving user data from said user;

receiving a save command from said user to save said user data; and

in response to receiving said save command, dynamically creating a new hypertext document containing said user data and displaying a message prompting the user to save the new document using said function for locally saving displayed documents, said new hypertext document containing a script function that becomes active when said new hypertext document is loaded to perform a desired restoration function.

2.   The method of claim 1 in which said client application receives said first hypertext document from a server application.

3.   The method of claim 1 in which said hypertext documents are HTML documents.

4.   The method of claim 1 in which said message is created as a part of said new hypertext document.

5.   The method of claim 1, comprising the further step of:

receiving a restore command from said user to restore previously saved user data; and

in response to receiving said restore command, repopulating said first document with said previously saved user data.

6.   In an information handling system in which a client application displays a first hypertext document to a user for entry of user data, said client application having a function for locally saving displayed documents, apparatus for providing local data persistence for said client application, said apparatus being associated with said client application and comprising:

means for receiving user data from said user;

means for receiving a save command from said user to save said user data; and

means responsive to receiving said save command for dynamically creating a new hypertext document containing said user data and displaying a message prompting the user to save the new document using said function for locally saving displayed documents, said new hypertext document containing a script function that becomes active when said new hypertext_ document is loaded to perform a desired restoration function.

7.   The apparatus of claim 6 in which said message is created as a part of said new hypertext document.

8.   The apparatus of claim 6, further comprising:

means for receiving a restore command from said user to restore previously saved user data; and

means responsive to receiving said restore command for repopulating said first document with said previously saved user data.

9.   A program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform method steps for providing local data persistence for a client application in an information handling system in which a client application displays a first hypertext document to a user for entry of user data, said client application having a function for locally saving displayed documents, said method steps comprising:

receiving user data from said user;

receiving a save command from said user to save said user data; and

in response to receiving said save command, dynamically creating a new hypertext document containing said user data and displaying a message prompting the user to save the new

document using said function for locally saving displayed documents, said new hypertext document containing a script function that becomes active when said new hypertext document is loaded to perform a desired restoration function.

10.   The program storage device of claim 9 in which said message is created as a part of said new hypertext document.

11.   The program storage device of claim 9, comprising the further step of:
          receiving a restore command from said user to restore previously saved user data; and
          in response to receiving said restore command, repopulating said first document with said previously saved user data.

12.   The method of claim 1 in which said script function is a JavaScript function.

13.   The method of claim 1 in which said script function becomes active when loaded to repopulate the first hypertext document with said user data.

14.   The apparatus of claim 6 in which said script function is a JavaScript function.

15.   The apparatus of claim 6 in which said script function becomes active when loaded to repopulate the first hypertext document with said user data.

16.   The program storage device of claim 9 in which said script function is a JavaScript function.

17.   The program storage device of claim 9 in which said script function becomes active when loaded to repopulate the first hypertext document with said user data.

EVIDENCE APPENDIX

(None)

RELATED PROCEEDINGS APPENDIX

(None)